FPT Approximation for Constrained Metric k-Median/Means

Dishant Goyal
Indian Institute of Technology Delhi, India
Dishant.Goyal@cse.iitd.ac.in

Ragesh Jaiswal[1]
Indian Institute of Technology Delhi, India
rjaiswal@cse.iitd.ac.in

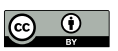
Amit Kumar
Indian Institute of Technology Delhi, India
amitk@cse.iitd.ac.in

Abstract

The Metric k-median problem over a metric space $(\mathcal{X}, d)$ is defined as follows: given a set $L \subseteq \mathcal{X}$ of *facility locations* and a set $C \subseteq \mathcal{X}$ of *clients*, open a set $F \subseteq L$ of k facilities such that the total service cost, defined as $\Phi(F, C) \coloneqq \sum_{x \in C} \min_{f \in F} d(x, f)$, is minimised. The metric k-means problem is defined similarly using squared distances (i.e., $d^2(.,.)$ instead of $d(.,.)$). In many applications there are additional constraints that any solution needs to satisfy. For example, to balance the load among the facilities in resource allocation problems, a capacity u is imposed on every facility. That is, no more than u clients can be assigned to any facility. This problem is known as the *capacitated* k-means/k-median problem. Likewise, various other applications have different constraints, which give rise to different *constrained* versions of the problem such as *r-gather*, *fault-tolerant*, *outlier* k-means/k-median problem. Surprisingly, for many of these constrained problems, no constant-approximation algorithm is known. Moreover, the *unconstrained* problem itself is known [1] to be W[2]-hard when parameterized by k. We give FPT algorithms with constant approximation guarantee for a range of constrained k-median/means problems. For some of the constrained problems, ours is the first constant factor approximation algorithm whereas for others, we improve or match the approximation guarantee of previous works. We work within the unified framework of Ding and Xu [24] that allows us to simultaneously obtain algorithms for a range of constrained problems. In particular, we obtain a $(3 + \varepsilon)$-approximation and $(9 + \varepsilon)$-approximation for the constrained versions of the k-median and k-means problem respectively in FPT time. In many practical settings of the k-median/means problem, one is allowed to open a facility at any client location, i.e., $C \subseteq L$. For this special case, our algorithm gives a $(2 + \varepsilon)$-approximation and $(4 + \varepsilon)$-approximation for the constrained versions of k-median and k-means problem respectively in FPT time. Since our algorithm is based on simple sampling technique, it can also be converted to a constant-pass log-space streaming algorithm. In particular, here are some of the main highlights of this work:

1. For the uniform capacitated k-median/means problems our results matches previously known results of Addad et al. [19].
2. For the r-gather k-median/means problem (clustering with lower bound on the size of clusters), our FPT approximation bounds are better than what was previously known.
3. Our approximation bounds for the fault-tolerant, outlier, and uncertain versions is better than all previously known results, albeit in FPT time.
4. For certain constrained settings such as chromatic, l-diversity, and semi-supervised k-median/means, we obtain the first constant factor approximation algorithms to the best of our knowledge.
5. Since our algorithms are based on a simple sampling based approach, we also obtain constant-pass log-space streaming algorithms for most of the above-mentioned problems.

[1] Part of this work was done while the author was on a sabbatical from IIT Delhi and visiting UC San Diego.



15th International Symposium on Parameterized and Exact Computation (IPEC 2020).
Editors: Yixin Cao and Marcin Pilipczuk; Article No. 14; pp. 14:1–14:19

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Facility location and clustering

Keywords and phrases k-means, k-median, approximation algorithms, parameterised algorithms

Digital Object Identifier 10.4230/LIPIcs.IPEC.2020.14

Related Version A full version of the paper is available at `https://arxiv.org/abs/2007.11773`.

Acknowledgements The authors would like to thank Anup Bhattacharya for useful discussions. Dishant Goyal would like to thank TCS Research Scholar Program.

1 Introduction

The metric k-means and k-median problems are similar. We combine the discussion of these problems by giving a definition of the k-service problem that encapsulates both these problems.

▶ **Definition 1** (k-service problem). *Let $(\mathcal{X}, d)$ be a metric space, $k > 0$ be any integer and $\ell \geq 0$ be any real number. Given a set $L \subseteq \mathcal{X}$ of feasible facility locations, and a set $C \subseteq \mathcal{X}$ of clients, find a set $F \subseteq L$ of k facilities that minimises the total service cost: $\Phi(F, C) \equiv \sum_{j \in C} \min_{i \in F} d^{\ell}(i, j)$.*

Note that the k-service problem is also studied with respect to a more general cost function $\sum_{j \in C} \min_{i \in F} \delta(i, j)$, where $\delta(i, j)$ denotes the cost of assigning a client $j \in C$ to a facility $i \in F$. We consider the special case $\delta(i, j) \equiv d^{\ell}(i, j)$. For $\ell = 1$, the problem is known as the *k-median* problem and for $\ell = 2$, the problem is known as the *k-means* problem. The above definition is motivated by the *facility location* problem [43] and differs from it in two ways. First, in the facility location problem, one is allowed to open any number of facilities. Second, one has to pay for an additional facility establishment cost for every open facility. Thus the k-service problem is basically the facility location problem for a fixed number of facilities and 0 facility establishment costs.

The k-service problem can also be viewed as a clustering problem, where the goal is to group the objects that are similar to each other. Clustering algorithms are commonly used in data mining, pattern recognition, and information retrieval [33]. However, the notion of a cluster differs for different applications. For example, some applications consider a cluster as a dense region of points in the data-space [25, 5], while others consider it as a highly connected subgraph of a graph [32]. Likewise, various models have been developed in the past that capture the clustering properties in different ways [47]. The k-means and k-median problems are examples of the *center-based* clustering model. In this model, the objects are mapped to the points in a metric space such that the distance between the points captures the degree of dissimilarity between them. In other words, the closer the two points are, the more similar they are to each other. In order to measure the quality of a clustering, a center (known as the cluster representative) is assigned to each cluster and the cost is measured based on the distances of the points to their respective cluster centers. Then the problem objective is to obtain a clustering with the minimum cost. To view the k-median instance as a clustering instance, consider the client set as a set of data points and the facility locations as the feasible centers. In a feasible solution, the clients which are assigned to the same facility are considered a part of the same cluster and the corresponding facility act as their cluster center. During our discussion, we will use the term *center* and *facility* interchangeably. Similarly, we can view the k-means problem as a clustering problem where the cost is measured with respect to the squared distances.

Various variants of the k-median/means problem have been studied in the clustering literature. For example, the Euclidean k-means problem (where $C \subseteq L = \mathbb{R}^d$) is NP-hard even for a fixed k or a fixed dimension d [22, 4, 41, 45]. This opens the question of designing a PTAS (polynomial-time approximation schemes) for the problem when either the number of clusters or the dimension is fixed. Indeed, various PTASs are known under such conditions [38, 26, 15, 35, 27, 18]. In general, it is known that the problem can not be approximated within a particular constant factor, unless $\mathsf{P} = \mathsf{NP}$ [8, 16].

The hardness results in the previous paragraph was for Euclidean setting. These problems may be harder in general metric spaces which is indeed what has been shown. The metric k-median problem is hard to approximate within a factor strictly smaller than $(1 + 2/e)$, and the metric k-means problem is hard to approximate within a factor strictly smaller than $(1 + 8/e)$ [29, 34]. On the positive side, various constant-factor approximation algorithms are known for the k-means (and k-median) problems in the metric and Euclidean settings [36, 14, 7, 30, 40, 11, 3]. Improving these bounds is not the goal of this paper. Instead, we undertake the task of improving/obtaining approximation bounds of a more general class of problems called the *constrained k-means/k-median* problem. Let us see what these problems are and why they are important.

For many real-world applications, the classical (unconstrained) k-means and k-median problems do not entirely capture the desired clustering properties. For example, consider the popular *k-anonymity* principle [44]. The principle provides anonymity to a public database while keeping it meaningful at the same time. One way to achieve this is to cluster the data in such a way to release only partial information related to the clusters obtained. Further, to protect the data from the *re-identification* attacks, the clustering should be done in such a way that each cluster gets at least r data-points. This method is popularly known as *r-gather* clustering [2] (see the formal definition in Table 1). Likewise, various other applications impose a specific set of constraints on the clusters. Such applications have been studied extensively. A survey on these applications is mentioned in Section 1.1 of [24]. We collectively mention these problems in Table 1 and their known approximation results in Table 2. We discuss these problems and their known results in detail in the full version of the paper.

An important distinction between the constrained problems and their unconstrained counterparts is the idea of *locality*. In simple words, the *locality* property says that the points which are close to each other should be part of the same cluster. This property holds for the unconstrained version of the problem. However, this may not necessarily hold for many of the constrained versions of the problem where minimising clustering cost is not the only requirement. To understand this, consider a center-set $F = \{f_1, f_2, \ldots, f_k\}$ and let $\{C_1, ..., C_k\}$ denote the clustering of the dataset such that the cost function gets minimised. That is, C_i contain all the points for which f_i is the closest center in the set F. Note that the clustering $\{C_1, ..., C_k\}$ just minimises the distance based cost function and may not satisfy any additional constraint that the clustering may need to satisfy in a constrained setting. In a constrained setting we may need an algorithm that, given a center-set $\{f_1, ..., f_k\}$ as input, outputs a clustering $\{\bar{C}_1, ..., \bar{C}_k\}$ which in addition to minimising $\sum_i \sum_{x \in \bar{C}_i} d^\ell(x, f_i)$ also satisfies certain clustering constraints. Such an algorithm is called a *partition algorithm*. In the unconstrained setting, the partition algorithm simply assigns points to closest center in F. However, designing such an efficient partition algorithm for the constrained versions of the problem is a non-trivial task. Ding and Xu [24] gave partition algorithms for all the problems mentioned in Table 1 (see Section 4 and 5.3 of [24]). Though these algorithms were specifically designed for the Euclidean space, they can be generalized to any metric space. We will see that such a partition algorithm is crucial in the design of our FPT algorithms.

Table 1 Constrained k-service problems with efficient partition algorithm (see Section 4 and 5.3 in [24] and references therein). The (*) marked problems were not discussed in [24]. Note that for problems 1 and 2, the bounds are cluster-wise and not facility-wise. Please see definition of $\Psi(F, \xi)$ and $\Psi^*(\xi)$ below (see eqn. (1) and defn. 2).

#	Problem	Description
1.	r-gather k-service problem* (r,k)-`GService`	Find clustering $\xi = \{C_1, ..., C_k\}$ with minimum $\Psi^*(\xi)$ such that for all i, $\lvert C_i\rvert \geq r_i$
2.	r-Capacity k-service problem* (r,k)-`CaService`	Find clustering $\xi = \{C_1, ..., C_k\}$ with minimum $\Psi^*(\xi)$ such that for all i, $\lvert C_i\rvert \leq r_i$
3.	l-Diversity k-service problem (l,k)-`DService`	Given that every client has an associated colour, find a clustering $\xi = \{C_1, ..., C_k\}$ with minimum $\Psi^*(\xi)$ such that for all i, the fraction of points sharing the same colour inside C_i is $\leq \frac{1}{l}$
4.	Chromatic k-service problem k-`ChService`	Given that every client has an associated colour, find a clustering $\xi = \{C_1, ..., C_k\}$ with minimum $\Psi^*(\xi)$ such that for all i, C_i should not have any two points with the same colour.
5.	Fault tolerant k-service problem (l,k)-`FService`	Given a value l_p for every client, find a clustering $\xi = \{C_1, ..., C_k\}$ and a set F of k centers, such that the sum of service cost of the points to l_p of nearest centers out of $F = \{f_1, f_2, \ldots, f_k\}$, is minimised.
6.	OWA k-service problem* k-`OWAService`	Given a vector $(w_1, ..., w_k)$ of non-increasing weights, find a center set $\{f_1, ..., f_k\}$ such that $\sum_{x \in C} \sum_{j=1}^{k} w_j \cdot \left(\overline{d}_j(x)\right)^{\ell}$ is minimised. Here, $\left(\overline{d}_1(x), ..., \overline{d}_k(x)\right)$ is a non-decreasing ordering of $\left(d(x, f_1), ..., d(x, f_k)\right)$.
7.	Semi-supervised k-service problem k-`SService`	Given a target clustering $\xi' = \{C'_1, ..., C'_k\}$ and constant α, find a clustering $\xi = \{C_1, ..., C_k\}$ and a center set F, such that the cost $\overline{\Psi}(F, \xi) \coloneqq \alpha \cdot \Psi(F, \xi) + (1 - \alpha) \cdot Dist(\xi', \xi)$ is minimised. $Dist$ denotes the set-difference distance.
8.	Uncertain k-service problem k-`UService`	Given a discrete probability distribution for every client, i.e., for a point $p \in C$ there is a set $D_p = \{p_1, \ldots, p_h\}$ such that p takes the value p_i with probability t_p^i and $\sum_{i=1}^{h} t_p^i \leq 1$. Find a clustering $\xi = \{C_1, ..., C_k\}$ so that the expected cost of $\Psi^*(\xi)$ is minimized.
9.	Outlier k-service problem* (k,m)-`OService`	Find a set $Z \subseteq C$ of size m and a clustering $\mathcal{C}' = \{C'_1, ..., C'_k\}$ of the set $C' \coloneqq C \setminus Z$, such that $\Psi^*(\xi')$ is minimized.

The partition algorithm gives us a way for going from center-set to clustering. What about the reverse direction? Given a clustering $\xi = \{C_1, C_2, \ldots, C_k\}$, can we find a center set that gives minimum clustering cost? The solution to this problem is simple. Construct a complete weighted bipartite graph $G = (V_l, V_r, E)$, where a vertex in V_l corresponds to a facility location in L, and a vertex in V_r corresponds to a cluster $C_j \in \xi$. The weight on an edge $(i, j) \in V_l \times V_r$ is equal to the cost of assigning the cluster C_j to the i^{th} facility, i.e., $\sum_{x \in C_j} d^{\ell}(x, i)$. Then we can easily obtain an optimal assignment by finding the minimum cost perfect matching in the graph G. Let us denote the minimum cost by $MCPM(\xi, L)$ Thus, it is sufficient to output an optimal clustering for a constrained k-service instance. In fact, all problems in Table 1 only requires us to output an optimal clustering for the problem.

Ding and Xu [24] suggested the following unified framework for considering any constrained k-means/k-median problem by modelling an arbitrary set of constraints using feasible clusterings. Note that they studied the problem in the Euclidean space where $C \subseteq L = \mathbb{R}^d$ whereas we study the problem in general metric space where L and C are discrete and

separate sets. We will use a few more definitions to define the problem. A k-center-set is a set of k distinct elements from L and for any k-center-set $F = \{f_1, ..., f_k\}$ and a clustering $\xi = \{C_1, ..., C_k\}$, we will use the cost function:

$$\Psi(F, \xi) \equiv \min_{\text{permutation } \pi} \left\{ \sum_{i=1}^{k} \sum_{x \in C_i} d^{\ell}(x, f_{\pi(i)}) \right\}. \tag{1}$$

▶ **Definition 2** (Constrained k-service problem)**.** *Let $(\mathcal{X}, d)$ be a metric space, $k > 0$ be any integer and $\ell \geq 0$ be any real number. Given a set $L \subseteq \mathcal{X}$ of feasible facility locations, a set $C \subseteq \mathcal{X}$ of clients, and a set $\mathbb{C}$ of feasible clusterings, find a clustering $\xi = \{C_1, C_2, \ldots, C_k\}$ in $\mathbb{C}$, that minimizes the following objective function: $\Psi^*(\xi) \equiv \min_{k\text{-center-set } F} \Psi(F, \xi)$.*

Note that $\Psi^*(\xi)$ is $MCPM(\xi, L)$, the minimum cost perfect matching as discussed earlier. The key component of the above definition is the set of feasible clusterings $\mathbb{C}$. Using this, we can define any constrained version of the problem. Note that $\mathbb{C}$ can have an exponential size. However, for many problems it can be defined concisely using a simple set of mathematical constraints. For example, $\mathbb{C}$ for the r-gather problem can be defined as $\mathbb{C} := \{\xi \mid \text{for every cluster } C_i \in \xi, |C_i| \geq r_i\}$, where $\xi = \{C_1, C_2, \ldots, C_k\}$ is a partitioning of the client set. Note that we consider the *hard assignment* model for the problem. That is, one cannot open more than one facility at a location. It differs from the *soft assignment* model where one can open multiple facilities at a location. The soft version can be stated in terms of the hard version – by allowing L to be a multi-set and creating k-copies for each location in L. It has been observed that the soft-assignment models are easier and allow better approximation guarantees than the hard-assignment models [21, 39]. For our discussion, we will call a center-set a *soft center-set* if it contains facility location multiple times, otherwise we call it a *hard center-set*. In fact, a soft center-set is a multi-set. We will avoid using the term multi-set to keep our discussion simple.

As observed in past works [24, 9], any constrained version of k-median/means can be solved using a partition algorithm for this version and a solution to a very general "list" version of the clustering problem which we discuss next. Let us define this problem which we call the *list k-service* problem [2]. This will help us solve the constrained k-service problem.

▶ **Definition 3** (List k-service problem)**.** *Let α be a fixed constant. Let $\mathcal{I} = (L, C, k, d, \ell)$ be any instance of the k-service problem and let $\xi = \{C_1, C_2, \ldots, C_k\}$ be an arbitrary clustering of the client set C. The goal of the problem is: given $\mathcal{I}$, find a list $\mathcal{L}$ of k-center-sets (i.e., each element of the list is a set of k distinct elements from L) such that, with probability at least $(1/2)$, $\mathcal{L}$ contains a k-center-set F such that $\Psi(F, \xi) \leq \alpha \cdot \Psi^*(\xi)$.*

Note that the clustering algorithm in the above setup does not get access to the clustering ξ and yet is supposed to find good centers (constant α approximation) for this clustering. Given this, it is easy to see that finding a single set of k centers that are good for ξ is not possible. However, finding a reasonably small list of k-center-sets such that at least one of the k-center-sets in the list is good may be feasible. This is main realization behind the formulation of the list version of the problem. The other reason is that since the target clustering is allowed to be a completely arbitrary partition of the client set C, we can use the solution of the list k-service problem to solve any constrained k-service problem as

[2] This notion of list version of the clustering problem was implicitly present in the work of Ding and Xu [24]. Bhattacharya et al. [9] formalized this as the *list k-means problem*.

long as there is a partition algorithm. The following theorem combines the list k-service algorithm and the partition algorithm for a constrained version of the problem to produce a constant-approximation algorithm this problem.

▶ **Theorem 4.** *Let $\mathcal{I} = (L, C, k, d, \ell, \mathbb{C})$ be any instance of any constrained k-service problem and let $A_{\mathbb{C}}$ be the corresponding partition algorithm. Let B be an algorithm for the list k-service problem that runs in time T_B for instance (L, C, k, d, ℓ). There is an algorithm that, with probability at least $1/2$, outputs a clustering $\xi \in \mathbb{C}$, which is an α-approximation for the constrained k-service instance. The running time of the algorithm is $O(T_B + |\mathcal{L}| \cdot T_A)$, where T_A is the running time of the partition algorithm.*

Proof. The algorithm is as follows. We first run algorithm B to obtain a list $\mathcal{L}$. For every k-center-set in the list, the algorithm runs the partition algorithm $A_{\mathbb{C}}$ on it. Then the algorithm outputs that k-center-set that has the minimum clustering cost. Let F' be this k-center-set and ξ' be the corresponding clustering. We claim that (F', ξ') is an α-approximation for the constrained k-service problem with probability at least $1/2$.

Let ξ^* be an optimal solution for the constrained k-service instance $(L, C, k, d, \ell, \mathbb{C})$ and F^* denote the corresponding k-center-set. By the definition of the list k-service problem, with probability at least $1/2$, there is a k-center-set F in the list $\mathcal{L}$, such that $\Psi(F, \xi^*) \leq \alpha \cdot \Psi(F^*, \xi^*)$. Let $\xi = A_{\mathbb{C}}(F) \in \mathbb{C}$ be the clustering corresponding to F. Thus, $\Psi(F, \xi) \leq \Psi(F, \xi^*)$ and so with probability at least $1/2$, $\Psi(F, \xi) \leq \alpha \cdot \Psi(F^*, \xi^*)$. Since F' gives the minimum cost clustering in the list, we have $\Psi(F', \xi') \leq \Psi(F, \xi)$. Therefore, with probability at least $1/2$, $\Psi(F', \xi') \leq \alpha \cdot \Psi(F^*, \xi^*)$.

Since, the algorithm runs a partition procedure for every center set in the list, the running time of this step is $|\mathcal{L}| \cdot T_A$. Picking a minimum cost clustering from the list takes $O(|\mathcal{L}|)$ time. Hence the overall running time is $O(T_B + |\mathcal{L}| \cdot T_A)$. ◀

Now suppose we are given a list $\mathcal{L}$ of size $g(k)$ (for some function g) and a partition algorithm for the problem with FPT running time. Then by Theorem 4, we get an FPT algorithm for the constrained k-service problem. Since for many of the constrained k-service problems there exists efficient partition algorithms, it makes sense to design an algorithm for the list k-service problem that outputs a list of size at most $g(k)$. We design such an algorithm in this paper. We also need to make sure that the partition algorithms for constrained problems that we saw in Table 1 exist and our plan of approaching the constrained problem using the list problem can be executed. Indeed, Ding and Xu [24] gave partition algorithms for a number of constrained problems. We make addition to their list which allows us to discuss new problems in this work. These additions and other discussions on approaching specific constrained problems using the list problem is discussed in the full version of the paper. What we note here is that the approximation guarantee for the list problem carries over to all the constrained problem in Table 1. We now look at our main results for the list k-service problem and its main implications for the constrained problems.

1.1 Our Results

We will show the following result for the list k-service problem.

▶ **Theorem 5** (Main Theorem). *Let $0 < \varepsilon \leq 1$ and $\ell \geq 1$. Let (L, C, k, d, ℓ) be any k-service instance and let $\xi = \{C_1, C_2, \ldots, C_k\}$ be any arbitrary clustering of the client set. There is an algorithm that, with probability at least $1/2$, outputs a list $\mathcal{L}$ of size $(k/\varepsilon)^{O(k\,\ell^2)}$, such that there is a k-center-set $S \in \mathcal{L}$ in the list such that $\Psi(S, \xi) \leq (3^\ell + \varepsilon) \cdot \Psi^*(\xi)$. Moreover, the running time of the algorithm is $O\left(n \cdot (k/\varepsilon)^{O(k\,\ell^2)}\right)$. For the special case when $C \subseteq L$, the algorithm gives a $(2^\ell + \varepsilon)$-approximation guarantee.*

Using the above Theorem together with Theorem 4, we obtain the following main results for the constrained k-means and k-median problems.

▶ **Corollary 6** (k-means). *For any constrained version of the metric k-means problem with a partition algorithm with FPT running time $g(k) \cdot n^{O(1)}$, there is a $(9+\varepsilon)$-approximation algorithm with an FPT running time $g(k) \cdot (k/\varepsilon)^{O(k)} \cdot n^{O(1)}$. For a special case when $C \subseteq L$, the algorithm gives a $(4+\varepsilon)$-approximation guarantee.*

▶ **Corollary 7** (k-median). *For any constrained version of the metric k-median problem with a partition algorithm with FPT running time $g(k) \cdot n^{O(1)}$, there is a $(3+\varepsilon)$-approximation algorithm with an FPT running time of $g(k) \cdot (k/\varepsilon)^{O(k)} \cdot n^{O(1)}$. For a special case when $C \subseteq L$, the algorithm gives a $(2+\varepsilon)$-approximation guarantee.*

Note that by Theorem 4, as long as the running time of the partition algorithm is $g(k) \cdot n^{O(1)}$, the total running time of the algorithm still stays FPT. All the problems in Table 1 either have an efficient partition algorithm (polynomial in n and k) or a partition algorithm with an FPT running time. We discuss these partition algorithms in the full version of the paper. It should be noted that other than the problems mentioned in Table 1, our algorithm works for any problem that fits the framework of the constrained k-service problem (i.e., Definition 2) and has a partition algorithm. This makes the approach extremely versatile since one may be able to solve more problems that may arise in the future.[3] The known results on constrained problems in Table 1 is summarised in Table 2. Note that for **all** these problems we obtain FPT time $(9+\varepsilon)$-approximation and $(3+\varepsilon)$-approximation for k-means and k-median respectively. For the special case when $C \subseteq L$ (a facility can be opened at any client location), we obtain FPT time $(4+\varepsilon)$-approximation and $(2+\varepsilon)$-approximation for k-means and k-median respectively. There are some subtle differences in the problems in Table 1 and Table 2. This is to be able to compare our results with known results. We will highlight these differences in the related work section.

Moreover, we can convert our algorithms to streaming algorithms using the technique of Goyal et al. [28]. We basically require a streaming version of our algorithm for the list k-service problem and a streaming partition algorithm for the constrained k-service problem. In Section 1.5, we will design a constant-pass log-space streaming algorithm for the list k-service problem. We already know streaming partition algorithms for the various constrained k-service problems [28]. This would give a streaming algorithm for all the problems given in Table 1 except for the l-diversity and chromatic k-service problems. Although *single*-pass streaming algorithms are considered much useful, it is interesting to know that there is a constant-pass streaming algorithm for many constrained versions of the k-service problem.

1.2 Related Work

A unified framework for constrained k-means/k-median problems was introduced by Ding and Xu [24]. Using this framework, they designed a PTAS (fixed k) for various constrained clustering problems. However, their study was limited to the Euclidean space where $C \subseteq L = \mathbb{R}^d$. Their results were obtained through an algorithm for the list version of the k-means problem (even though it was not formally defined in their work). The running time of this algorithm was $O(nd \cdot (\log n)^k \cdot 2^{poly(k/\varepsilon)})$ and the list size was $(\log n)^k \cdot 2^{poly(k/\varepsilon)}$. Bhattacharya et al. [9] formally defined and studied the list k-service problem. They obtained a faster

[3] We note that new ways of modelling fairness in clustering is giving rise to new clustering problems with fairness constraints and some of these new problems may fit into this framework.

Table 2 Known results for the constrained clustering problems. Note that for **all** the above problems we obtain FPT time $(3+\varepsilon)$-approximation and $(9+\varepsilon)$-approximation for k-median and k-means respectively. For the special case when $C \subseteq L$ (a facility can be opened at any client location), we obtain FPT time $(2+\varepsilon)$-approximation and $(4+\varepsilon)$-approximation for k-median and k-means respectively. Note that uniform case for problems 1 and 2 means that the lower/upper bound on the size of all clusters is the same.

#	**Problem**	**Metric k-median**	**Metric k-means**
1.	r-gather k-service (uniform case)	7.2-approx [23] (for $C = L$) (in FPT time)	86.9-approx [23] (for $C = L$) (in FPT time)
2.	r-Capacity k-service (uniform case)	$(3+\varepsilon)$-approx [19] (in FPT time)	$(9+\varepsilon)$-approx [19] (in FPT time)
3.	l-Diversity k-service	-	-
4.	Chromatic k-service	-	-
5.	Fault tolerant k-service	93-approx. [31]	-
6.	OWA k-service	93-approx. [12]	-
7.	Semi-supervised k-service	-	-
8.	Uncertain k-service (assigned version)	$(6.35+\varepsilon)$-approx. [20] (for $C \subseteq L$)	$(74+\varepsilon)$-approx. [20] (for $C \subseteq L$)
9.	Outlier k-service	$(7+\varepsilon)$-approx. [37]	$(53+\varepsilon)$-approx. [37]

For the Euclidean k-means and k-median (where $C \subseteq L = \mathbb{R}^d$), all the constrained problems have an FPT time $(1+\varepsilon)$ approximation algorithm [24, 9].

algorithm for the list problem with running time to $O(nd \cdot (k/\varepsilon)^{O(\log(k/\varepsilon))})$ and list size to $(k/\varepsilon)^{O(\log(k/\varepsilon))}$ for the constrained k-means/k-median problem. Recently, Goyal et al. [28] obtained useful generalisations of the results of Bhattacharya et al. [9] and used this to design logspace (assuming k and ε are constants) streaming algorithms for various constrained versions of the problem. In this paper, we study the constrained k-means/median problems in general metric spaces while treating L and C as separate sets. More importantly, we design an algorithm that gives a better approximation guarantee than the previously known algorithms by taking advantage of FPT running time. Moreover, for many problems, it is the first algorithm that achieves a constant-approximation in FPT running time. Please see Table 2 for the known results on the problem. Due to space restrictions, we have a detailed discussion on these problems in the full version of the paper.

In the introduction, we would specifically like to discuss the result of Addad et al. [19] for the capacitated k-service problem. Their definition of the capacitated k-service problem is different from the one mentioned in Table 1 that we are considering. Following is their definition of the capacitated k-service problem.

▶ **Definition 8** (Addad et al. [19]). *Given an instance $\mathcal{I} = (L, C, k, d, \ell)$ of the k-service problem and a capacity function $r : L \to \mathbb{Z}_+$, find a set $F \subseteq L$ of k facilities such that the assignment cost $\sum_{j \in C} \min_{i \in F} d^\ell(j, i)$ is minimized, and no more than r_i clients are assigned to a facility $i \in L$.*

Note that in the above definition, there is a capacity associated with every facility location in L whereas in our definition, capacities are associated with the k clusters. This means that a facility can service arbitrary number of clients as long as the cluster sizes are bounded. This is not allowed as per the problem definition of Addad et al. [19]. However, for the uniform capacities the problem definitions are equivalent and the results become comparable. We match the approximation guarantees obtained Addad et al. [19] for the uniform case even though using very different techniques.

As we mentioned earlier, the unconstrained metric k-median problem is hard to approximate within a factor strictly smaller than $(1+2/e)$, and the metric k-means problem is hard to approximate within a factor strictly smaller than $(1+8/e)$. Surprisingly this lower bound persists even if we allow an FPT running time [17, 42]. However, this FPT lower bound is based on a complexity theoretic conjecture known as Gap-ETH [13]. The problem also has a matching upper bound algorithm with an FPT running time [17]. So, the unconstrained k-means and k-median problems in the metric setting is fairly well understood. On the other hand, our understanding of most constrained versions of the problem is still far from complete. We believe that our work is an important step in understanding constrained problems in general metric spaces.

1.3 Our Techniques

In this section, we discuss our sampling based algorithm for list k-service problem. First, let us define a few notations and identities that we will use often in our discussions. We define the unconstrained k-service cost of a set S with respect to a center set F as: $\Phi(F,S) := \sum_{x\in S} \min_{f\in F} d^\ell(f,x)$. For a singleton set $\{f\}$, we denote $\Phi(\{f\},S)$ by $\Phi(f,S)$. We denote the optimal (unconstrained) k-service cost of an instance (L,C,k,d,ℓ) by $OPT(L,C)$.

As described earlier, an FPT algorithm for the list k-service problem gives an FPT algorithm for a constrained version of the k-service problem that has an efficient or FPT-time partition algorithm. Given that we are allowed FPT running time for the list problem, it may be tempting to think of the following strategy: Use a bi-criteria approximation algorithm for the unconstrained version of the k-median problem to obtain $poly(k/\varepsilon)$ centers S and then use the partition algorithm on all k-sized subsets of S to pick the best one. Unfortunately, this strategy does not give a constant factor approximation. We discuss the details in the full version of the paper.

In this work, we give a sampling based algorithm that is similar to the algorithm of Goyal et al. [28] that was specifically designed for the Euclidean setting. However, working in a metric space instead of Euclidean space poses challenges as some of the main tools used for analysis in the Euclidean setting cannot be used in metric spaces. We carefully devise and prove new sampling lemmas that makes the high-level analysis of Goyal et al. [28] go through. Our algorithm is based on D^ℓ-sampling. Given a point set F, D^ℓ-sampling a point from the client set C w.r.t. center set F means sampling using the distribution where the sampling probability of a client $x \in C$ is $\frac{\Phi(F,\{x\})}{\Phi(F,C)} = \frac{\min_{f\in F} d^\ell(f,x)}{\sum_{y\in C}\min_{f\in F} d^\ell(f,y)}$. In case F is empty, then D^ℓ-sampling is the same as uniform sampling. Please see Algorithm 1 for the list k-service problem.

Let us discuss some of the main ideas of the algorithm and its analysis. First, note that as per the algorithm description, the list size is $2^k \cdot \binom{(\eta+1)k^2}{k}$ which is $(k/\varepsilon)^{O(k\ell^2)}$ for the parameters given. This is because in step (9), the algorithm considers all possible k sized subsets of (multi)set T of size $(\eta+1)k^2$. We now discuss the approximation guarantee. Note that in the first step, we obtain a center-set $F \subseteq C$ which is an α-approximation for the

Algorithm 1 Algorithm for the list k-service problem.

```
List-k-service (L, C, k, d, ℓ, ε)
    Inputs: k-service instance (L, C, k, d, ℓ) and accuracy ε
    Output: A list 𝓛, each element in 𝓛 being a k-center set
    Constants: β = 4^{ℓ−1} · ( ℓ^ℓ · 3^{ℓ²+4ℓ+3} / ε^{ℓ+1} + 1 ); γ = ℓ^ℓ · 3^{ℓ²+5ℓ+1} / ε^ℓ;
    η = α β γ k · 3^{ℓ+2} / ε²
(1)  Run any α-approximation algorithm with α = poly(k) for the unconstrained
     k-service instance (C, C, k, d, ℓ) and let F be the obtained center-set.
       (k-means++ [6] is one such algorithm.)
(2)  𝓛 ← ∅
(3)  Repeat 2^k times:
(4)      Sample a multi-set M of ηk points from C using D^ℓ-sampling w.r.t.
          center set F
(5)      M ← M ∪ F
(6)      T ← ∅
(7)      For every point x in M:
(8)          T ← T ∪ {k points in L that are closest to x}
(9)      For all subsets S of T of size k:
(10)         𝓛 ← 𝓛 ∪ {S}
(11) return(𝓛)
```

unconstrained k-service instance (C, C, k, d, ℓ). Any α that is polynomial in k suffices for our analysis. That is, $\Phi(F, C) \leq \alpha \cdot OPT(C, C)$. One such algorithm is the k-means++ algorithm [6] that gives an $O(4^\ell \cdot \log k)$-approximation guarantee and a running time $O(nk)$. Now, let us see how the center-set F can help us. Let us focus on any cluster C_i of a target clustering $\xi = \{C_1, \ldots, C_k\}$. We note that the closest facility to a uniformly sampled client from any client set C_i provides a constant approximation to the optimal 1-median/means cost for C_i in expectation. This is formalized in the next lemma. This lemma (or a similar version) has been used in multiple other works in analysing sampling based algorithms (for example, see Lemma 3.1 in [6]). This lemma is restated and formally proven in the full version of the paper.

▶ **Lemma 9.** *Let $S \subseteq C$ be any subset of clients and let f^* be any center in L. If we uniformly sample a point x in S and open a facility at the closest location in L, then the following identity holds:*

$$\mathbb{E}[\Phi(t(x), S)] \leq 3^\ell \cdot \Phi(f^*, S), \text{ where } t(x) \text{ is the closest facility location from } x.$$

Unfortunately, we cannot uniformly sample from C_i directly since C_i is not known to us. Given this, our main objective should be to use F to try to uniformly sample from C_i so that we could achieve a constant approximation for C_i. Let us do a case analysis based on the distance of points in C_i from the nearest point in F. Consider the following two possibilities: The first possibility is that the points in C_i are close to F. If this is the case, we can uniformly sample a point from F instead of C_i. This would incur some extra cost. However, the cost is small and can be bounded. To cover this first possibility, the algorithm adds the entire set F to the set of sampled points M (see line (5) of the algorithm). The second possibility

is that the points in C_i are far-away from F. In this case, we can D^ℓ-sample the points from C. Since the points in C_i are far away, the sampled set would contain a good portion of points from C_i and the points will be *almost* uniformly distributed. We will show that almost uniform sampling is sufficient to apply Lemma 9 on C_i. However, we would have to sample a large number of points to boost the success probability. This requirement is taken care of by line (4) of the algorithm. Note that we may need to use a hybrid approach for analysis since the real case may be a combination of the first and second possibility. Most of the ingenuity of this work lies in formulating and proving appropriate sampling lemmas to make this hybrid analysis work.

To apply Lemma 9, we need to fulfill one more condition. We need the closest facility location from a sampled point. This requirement is handled by lines (7) and (8) of the algorithm. However, note that the algorithm picks k-closest facility locations instead of just one facility location. We will show that this step is crucial to obtain a *hard-assignment* solution for the problem. Finally, the algorithm adds all the potential center sets to a list $\mathcal{L}$ (see line (9) and (10) of the algorithm). The algorithm repeats this procedure 2^k times to boost the success probability (see line (3) of the algorithm). We will show the following result from which our main theorem (Theorem 5) trivially follows.

▶ **Theorem 10.** *Let $0 < \varepsilon \leq 1$ and $\ell \geq 1$. Let (L, C, k, d, ℓ) be any k-service instance and let $\xi = \{C_1, C_2, \ldots, C_k\}$ be any arbitrary clustering of the client set. The algorithm* `List-k-service`*$(L, C, k, d, \ell, \varepsilon)$, with probability at least $1/2$, outputs a list $\mathcal{L}$ of size $(k/\varepsilon)^{O(k\,\ell^2)}$, such that there is a k center set $S \in \mathcal{L}$ in the list such that $\Psi(S, \xi) \leq (3^\ell + \varepsilon) \cdot \Psi^*(\xi)$. Moreover, the running time of the algorithm is $O\left(n \cdot (k/\varepsilon)^{O(k\,\ell^2)}\right)$. For the special case of $C \subseteq L$, the approximation guarantee is $(2^\ell + \varepsilon)$.*

The details of the analysis is given in the full version of the paper. Here, we give the high-level outline of the proof. Let $\xi = \{C_1, C_2, \ldots, C_k\}$ be the (unknown) target clustering and $F^* = \{f_1^*, f_2^*, \ldots, f_k^*\}$ be the corresponding optimal center set. Suppose C_i is assigned to f_i^*, and $\Delta(C_i)$ denote its corresponding cost, i.e., $\Delta(C_i) = \Phi(f_i^*, C_i)$. Let us classify the clusters into two categories: W and H.

$$W \coloneqq \{C_i \mid \Phi(F, C_i) \leq \frac{\varepsilon}{\alpha\,\gamma\,k} \cdot \Phi(F, C),\ \text{for } 1 \leq i \leq k\}$$

$$H \coloneqq \{C_i \mid \Phi(F, C_i) > \frac{\varepsilon}{\alpha\,\gamma\,k} \cdot \Phi(F, C),\ \text{for } 1 \leq i \leq k\}$$

In other words, W contains the *low-cost* clusters and H contains the *high-cost* clusters with respect to F. Now, let us look at the set M obtained by lines (4) and (5) of the algorithm. M contains some D^ℓ-sampled points from C and the center set F. We show that M has the following property.

Property-I: For any cluster $C_i \in \{C_1, C_2, \ldots, C_k\}$, with probability at least $1/2$, there is a point s_i in M such that the following holds:

$$\Phi(t(s_i), C_i) \leq \begin{cases} \left(3^\ell + \frac{\varepsilon}{2}\right) \cdot \Delta(C_i) + \frac{\varepsilon}{2^{\ell+1}\,k} \cdot OPT(C, C), & \text{if } C_i \in W \\ \left(3^\ell + \frac{\varepsilon}{2}\right) \cdot \Delta(C_i), & \text{if } C_i \in H \end{cases}$$

where $t(s_i)$ denotes any facility location that is closer to s_i than f_i^*, i.e., $d(s_i, t(s_i)) \leq d(s_i, f_i^*)$.

First, let us see how this property gives the desired result. By a well known fact, we have $OPT(C, C) \leq 2^{\ell} \cdot OPT(L, C)$. Moreover, the optimal cost $OPT(L, C)$ of the unconstrained k-service instance is always less than the constrained k-service cost $\sum_{i=1}^{k} \Delta(C_i)$. Therefore, Property-I implies that $T_s \coloneqq \{t(s_1), t(s_2), \ldots, t(s_k)\}$ is a $\left(3^{\ell} + \varepsilon\right)$-approximation for ξ, with probability at least $\frac{1}{2^k}$. [4] Now, note that the facility locations that are closest to s_i satisfy the definition of $t(s_i)$. Moreover, the algorithm adds one such facility location to set T (see line (8) of the algorithm). Thus there is a center-set T_s in the list that gives $(3^{\ell}+\varepsilon)$-approximation for ξ. To boost the success probability to $1/2$, the algorithm repeats the procedure 2^k times (see line (3) of the algorithm). Based on these arguments, it looks like we got the desired result. However, there is one issue that we need to take care of. Remember, we are looking for a hard assignment for the problem, and the set T_s could be a soft center-set, since the closest facility locations might be same for s_i's. In other words, $t(s_i)$ could be same as $t(s_j)$ for some $i \neq j$. At the end of this section we will show that there is indeed a hard center-set in the list $\mathcal{L}$, that gives the required approximation for the problem. For now let us try to argue Property-I for M and the target clusters. First consider the case of low-cost clusters as follows.

Case 1: $\boldsymbol{\Phi(F, C_i) \leq \frac{\varepsilon}{\alpha \gamma k} \cdot \Phi(F, C)}$

For a point $x \in \mathcal{X}$, let $c(x)$ denote the closest location in F. Based on this definition, consider a multi-set $M_i \coloneqq \{c(x) \mid x \in C_i\}$. Since C_i has a low cost with respect to F, the points in C_i are close to from points from F. Consider uniformly sampling a point from M_i. In the next lemma, we show that a uniformly sampled point from M_i is a good enough center for C_i. We give the proof in the full version of the paper.

▶ **Lemma 11.** *Let p be a point sampled uniformly at random from M_i. Then the following bound holds:*

$$\mathbb{E}[\Phi(t(p), C_i)] \leq \left(3^{\ell} + \frac{\varepsilon}{2}\right) \cdot \Delta(C_i) + \frac{\varepsilon}{2^{\ell+1} k} \cdot OPT(C, C).$$

Since the above lemma estimates the average cost corresponding to a sampled point, there has to be a point p in M_i such that $\Phi(t(p), C_i) \leq \left(3^{\ell} + \frac{\varepsilon}{2}\right) \cdot \Delta(C_i) + \frac{\varepsilon}{2^{\ell+1} k} \cdot OPT(C, C)$. Since M_i is only composed of the points from F and we keep the entire set F in M (see line (5) of the algorithm), therefore Property-I is satisfied for every cluster $C_i \in W$. Let us now prove Property I for the high cost clusters.

Case 2: $\boldsymbol{\Phi(F, C_i) > \frac{\varepsilon}{\alpha \gamma k} \cdot \Phi(F, C)}$

Since the cost of the cluster is high, some points of C_i are far away from the center set F. We partition C_i into two sets: C_i^n and C_i^f, as follows.

$$C_i^n \coloneqq \{x \mid d^{\ell}(c(x), x) \leq R^{\ell}, \text{ for } x \in C_i\}, \quad \text{where } R^{\ell} = \frac{1}{\beta} \cdot \frac{\Phi(F, C_i)}{|C_i|}$$

$$C_i^f \coloneqq \{x \mid d^{\ell}(c(x), x) > R^{\ell}, \text{ for } x \in C_i\}, \quad \text{where } R^{\ell} = \frac{1}{\beta} \cdot \frac{\Phi(F, C_i)}{|C_i|}$$

[4] Note that the probabilities can be multiplied since M can be partitioned into k groups and we actually show that the good point s_i for C_i is either in F or is in any group with probability at least $1/2$.

In other words, C_i^n represents the set of points that are *near* to the center set F and C_i^f represents the set of points that are *far* from the center set F. Recall that our prime objective is to obtain a uniform sample from C_i, so that we can apply lemma 9. To achieve that we consider sampling from C_i^n and C_i^f separately. The idea is as follow. To sample a point from C_i^f we use the D^ℓ-sampling technique and show that it gives an almost uniform sample from C_i^f. For C_i^n, we will use F as its proxy, and sample a point from F instead. However, doing so would incur an extra cost. Since we are using F as a proxy for C_i^n, we define a multi-set $M_i^n \coloneqq \{c(x) \mid x \in C_i^n\}$. Let us define another multi-set $M_i \coloneqq C_i^f \cup M_i^n$. In the following lemma we show that there is a point in M_i that is a good center for C_i. The lemma is similar to lemma 11 of the low-cost clusters. The formal proof is given in the full version of the paper.

▶ **Lemma 12.** *Let p be a point sampled uniformly at random from M_i. Then the following bound holds:*

$$\mathbb{E}[\Phi(t(p), C_i)] \leq \left(3^\ell + \frac{\varepsilon}{4}\right) \cdot \Delta(C_i)$$

The above lemma gives a bound on the expectation. To show that M has a good center for high-cost cluster C_i with high probability, we need to make sure that adequate samples are obtained in line (4) of the algorithm. The choice of parameters β, γ, and η is based on this probability analysis and is given in the full version of the paper.

Having argued that Property-I is satisfied for every cluster in W and H, we can finally claim that $T_s = \{t(s_1), t(s_2), \ldots, t(s_k)\}$ is a $\left(3^\ell + \varepsilon\right)$-approximation for ξ with probability at least $\frac{1}{2^k}$. However, as described earlier, T_s could be a soft center-set since $t(s_i)$ can be same as $t(s_j)$ for some $i \neq j$. To obtain a hard center-set, we make use of line (8) of the algorithm. In line (8), the algorithm pulls out the k closest points from L instead of just one. Note that it is not necessary to open a facility at a closest location in L. Rather, we can open a facility at any location f in L, that is at least as close to s_i as f_i^*, i.e., $d(s_i, f) \leq d(s_i, f_i^*)$.

Let $T(s_i)$ denote a set of k closest facility location for s_i. We show that there is a hard center-set $T_h \subset \cup_i T(s_i)$, such that $T_h \coloneqq \{f_1, \ldots, f_k\}$ and $d(s_i, f_i) \leq d(s_i, f_i^*)$ for every $1 \leq i \leq k$. We define T_h using the following simple subroutine:

```
1 FindFacilities
2   - T_h ← ∅
3   - For i ∈ {1, ..., k}:
4       - if (f_i^* ∈ T(s_i)) T_h ← T_h ∪ {f_i^*}
5       - else
6           - Let f ∈ T(s_i) be any facility such that f is not in T_h
7           - T_h ← T_h ∪ {f}
```

▶ **Lemma 13.** *$T_h = \{f_1, f_2, \ldots, f_k\}$ contains exactly k different facilities such that for every $1 \leq i \leq k$, we have $d(s_i, f_i) \leq d(s_i, f_i^*)$.*

Proof. First, let us show that all facilities in T_s are different. Since, f_i^* is different for different clusters, the if statement adds facilities in T_h that are different. In else part, we only add a facility to T_h that is not present in T_h. Thus the else statement also adds facilities in T_h that are different. Now, let us prove the second property, i.e., $d(s_i, f_i) \leq d(s_i, f_i^*)$ for every $1 \leq i \leq k$. The property is trivially true for the facilities added in the if statement. Now, for the facilities added in the second step we know that $T(s_i)$ does not contain f_i^*. Since, $T(s_i)$

is a set of k-closest facility locations, we can say that for any facility location f in $T(s_i)$, $d(s_i, f) \leq d(s_i, f_i^*)$. Thus any facility added in the else statement has $d(s_i, f) \leq d(s_i, f_i^*)$. This completes the proof. ◀

Thus $T_h \in \mathcal{L}$ is a hard center-set, which gives the $(3^\ell + \varepsilon)$-approximation for the problem. This completes the analysis of the algorithm.

Now, suppose we are given the flexibility to open a facility at a client location. In other words, suppose it is given that $C \subseteq L$. For this case, we can directly open the facilities at the locations $\{s_1, s_2, \ldots, s_k\}$ instead of $t(s_i)$'s, and we would not need lines (7) and (8) of the algorithm. Further, we can show that lemma 11 and 12 would give $(2^\ell + \varepsilon)$-approximation for this special case. However, please note that $\{s_1, s_2, \ldots, s_k\}$ is still a soft center-set. To obtain a hard center-set we do need to consider the k-closest facility locations for a point s_i. In that case, lemma 11 and 12 would not provide $(2^\ell + \varepsilon)$-approximation. Therefore, we need to make some changes in the analysis of lemma 11 and 12 to get back $(2^\ell + \varepsilon)$-approximation. We discuss these details in the full version of the paper.

1.4 A Matching Lower Bound on approximation

We gave sampling based algorithms and showed an approximation guarantee of $(3^\ell + \varepsilon)$ (and $(2^\ell + \varepsilon)$ for the special case $C \subseteq L$). In this subsection, we show that our analysis of the approximation factor is tight. More specifically, we will show that our algorithm does not provide better than $(3^\ell - \delta')$ approximation guarantee for arbitrarily small $\delta' > 0$ (and $2^\ell - \delta'$ for the case $C \subseteq L$). To show this, we create a *bad instance* for the problem in the following manner. We create the instance using an undirected weighted graph where $C \cup L$ is the vertex set of the graph and the shortest weighted path between two vertices defines the distance metric. The set C is partitioned into the subsets $C_1, C_2, \ldots, C_k$, and L is partitioned into the subsets $L_1, L_2, \ldots, L_k$. The subgraphs over $C_1 \cup L_1, C_2 \cup L_2, \ldots,$ and $C_k \cup L_k$ are all identical to each other. Let us describe the subgraph over vertex set $C_i \cup L_i$ in general. In this subgraph, all the clients are connected to a common facility location f_i^* with an edge of unit weight. Also, every client is connected to a distinct set of k facility locations with an edge of weight $(1 - \delta)$. We denote this set by $T(x)$ for a client $x \in C_i$. Figure 1 shows the complete description of this subgraph. Lastly, all pairs of subgraphs $C_i \cup L_i$ and $C_j \cup L_j$ are connected with an edge (f_i^*, f_j^*) of weight $\Delta \gg |C|$. This completes the construction of the bad instance.

Let us define a target clustering on the instance. Consider the unconstrained k-service problem. It is easy to see that $\xi = \{C_1, C_2, \ldots, C_k\}$ is an optimal clustering for this instance. The optimal cost of a cluster C_i is $\Phi(f_i^*, C_i) = |C_i|$, and the optimal cost of the entire instance is $OPT = \sum_i |C_i| = |C|$.

Now, we will show that any list $\mathcal{L}$ produced by the algorithm `List-k-service` does not contain any center-set that can provide better than $(3^\ell - \delta')$-approximation for ξ. To show this, let us examine every center-set in the list $\mathcal{L}$ produced by `List-k-service`. Note that the set T obtained in line (8) of the algorithm does not contain any optimal facility location f_i^* because f_i^* does not belong to $T(x)$. Therefore, no center set in the list contains any of the optimal facility locations $\{f_1^*, ..., f_k^*\}$. Let us evaluate the clustering cost corresponding to every center set in the list. Let $F = \{f_1, f_2, \ldots, f_k\}$ be a center-set in the list. We have two possibilities for the facilities in F. The first possibility is that, there are at least two facilities in F, that belongs to the same subgraph $C_i \cup L_i$. In this case, the cost of the target clustering is $\Psi(F, \xi) > \Delta \gg OPT$. So in this case, F gives an unbounded clustering cost. Let us consider the second possibility that all facilities in F belong to different subgraphs.

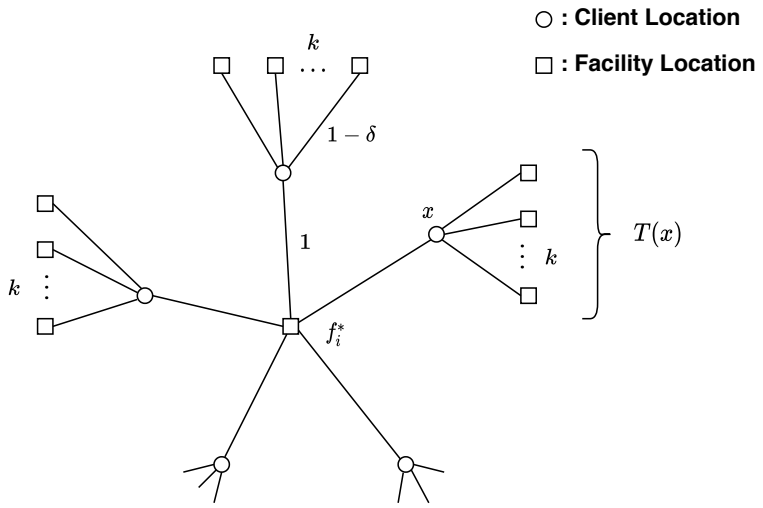


Figure 1 An undirected weighted subgraph on $C_i \cup L_i$.

Without loss of generality, we can assume that $f_i \in L_i$. Since f_i can not be the optimal facility location, we can further assume that $f_i \in T(x)$ for some $x \in C_i$. The cost of a cluster in this case is $\Phi(f_i, C_i) = (3-\delta)^\ell(|C_i|-1) + (1-\delta)^\ell > (3-\delta)^\ell(|C_i|-1)$. Hence, the overall cost of the instance is $\Psi(F,\xi) > (3-\delta)^\ell \cdot (|C|-k) \geq (3-\delta)^\ell \cdot |C| - 3^\ell\, k \geq (3^\ell - \delta') \cdot |C|$, for $\delta' = 3^{\ell-1} \cdot \ell\, \delta + \frac{3^\ell k}{|C|}$. Therefore, we can say that list does not contain any center set that can provide better than $(3^\ell - \delta')$ approximation guarantee for ξ.

▶ **Theorem 14.** *For any $0 < \delta' \leq 1$, there are instances of the k-service problem for which the algorithm* `List-k-service`$(L, C, k, d, \ell, \varepsilon)$ *does not provide better than $(3^\ell - \delta')$ approximation guarantee.*

Now, let us examine the same bad instance when we have the flexibility to open a facility at a client location. In this case, we have a third possibility that $F = \{f_1, f_2, \ldots, f_k\}$ such that f_i is some client location in C_i. The cost of a cluster in this case is $\Phi(f_i, C_i) = 2^\ell \cdot (|C_i| - 1)$ and the overall cost the instance is $\Psi(F,\xi) = 2^\ell \cdot |C| - 2^\ell \cdot k = (2^\ell - \delta') \cdot |C|$, for $\delta' = 2^\ell \cdot k/|C|$. So for the special case $C \subseteq L$, we obtain the following theorem.

▶ **Theorem 15.** *For any $0 < \delta' \leq 1$, there are instances of the k-service problem (with $C \subseteq L$), for which the algorithm* `List-k-service`$(L, C, k, d, \ell, \varepsilon)$ *does not provide better than $(2^\ell - \delta')$ approximation guarantee.*

1.5 Streaming Algorithms

In this subsection, we discuss how to obtain a constant-pass streaming algorithm using the ideas of Goyal et al. [28]. Our offline algorithm has two main components, namely: the list k-service algorithm and partition algorithm. The list k-service procedure is common to all constrained versions of the problem. However, the partition algorithm differs for different constrained versions. First, let us convert `List-k-service`(L, C, k, d, ℓ) algorithm to a streaming algorithm.

Algorithm Streaming algorithm.

1. In the first pass, we run a streaming α-approximation algorithm for the instance (C, C, k, d, ℓ). For this, we can use the streaming algorithm of Braverman et al. [10]. The algorithm gives a constant-approximation with the space complexity of $O(k \log n)$.
2. In the second pass, we perform the D^ℓ-sampling step using the *reservoir sampling* technique [46].
3. In the third pass, we find the k-closest facility locations for every point in M.

This gives us the following result.

▶ **Theorem 16.** *There is a 3-pass streaming algorithm for the list k-service problem, with the running time of $O(n \cdot f(k, \varepsilon))$ and space complexity of $f(k, \varepsilon) \cdot \log n$, where $f(k, \varepsilon) = (k/\varepsilon)^{O(k\ell^2)}$.*

Now, let us discuss the partition algorithms in streaming setting. For the l-diversity and chromatic k-service problems, it is known that there is no deterministic log-space streaming algorithm [28]. For the remaining constrained problems, there are streaming partition algorithms that are discussed in [28] (for the Euclidean setting) and the full version of the paper. Note that all of the streaming partitioning algorithms do not give an optimal partitioning but only a partitioning that is close to the optimal. Each algorithm makes at most 3-pass over the data-set and takes logarithmic space complexity. The partition algorithm, together with the list k-service algorithm, gives the following main results.

▶ **Theorem 17.** *For the following constrained k-service problems there is a 6-pass streaming algorithm that gives a $(3^\ell + \varepsilon)$-approximation guarantee: (1) r-gather k-service problem, (2) r-capacity k-service problem, (3) Fault-tolerant k-service problem, (4) Semi-supervised k-service problem, (5) Uncertain k-service problem (assigned case). The algorithm has the space complexity of $O(f(k, \varepsilon, \ell) \cdot \log n)$ and the running time of $O(f(k, \varepsilon, \ell) \cdot n^{O(1)})$, where $f(k, \varepsilon, \ell) = (k/\varepsilon)^{O(k\ell^2)}$. Further, the algorithm gives $(2^\ell + \varepsilon)$-approximation guarantee when $C \subseteq L$.*

▶ **Theorem 18.** *For the outlier k-service problem there is a 5-pass streaming algorithm that gives a $(3^\ell + \varepsilon)$-approximation guarantee. The algorithm has space complexity of $O(f(k, m, \varepsilon, \ell) \cdot \log n)$ and running time of $f(k, m, \varepsilon, \ell) \cdot n^{O(1)}$, where $f(k, m, \varepsilon, \ell) = ((k + m)/\varepsilon)^{O(k\ell^2)}$. Further, the algorithm gives $(2^\ell + \varepsilon)$-approximation guarantee when $C \subseteq L$.*

2 Conclusion and Open Problems

In this paper, we worked within the unified framework of Ding and Xu [24] to obtain simple sampling based algorithms for a range of constrained k-median/means problems in general metric spaces. Surprisingly, even working within this high-level framework, we obtained better (or matched) approximation guarantees of known results that were designed specifically for the constrained problem. On one hand, this shows the versatility of the unified approach along with the sampling method. On the other hand, it encourages us to try to design algorithms with better approximation guarantees for these constrained problems. Our matching approximation lower bound for the sampling algorithm suggests that further improvement may not be possible through sampling based ideas. On the lower bound side, it may be useful to obtain results similar to that for the unconstrained setting where approximation lower bounds of $(1 + 2/e)$ and $(1 + 8/e)$ are known for k-median and k-means respectively [17]. Another direction is to find other constrained problems that can fit into the unified framework and can benefit from the results in this work.

References

1 Marek Adamczyk, Jaroslaw Byrka, Jan Marcinkowski, Syed M. Meesum, and Michal Wlodarczyk. Constant-factor FPT approximation for capacitated k-median. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms (ESA 2019)*, volume 144 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ESA.2019.1.

2 Gagan Aggarwal, Rina Panigrahy, Tomás Feder, Dilys Thomas, Krishnaram Kenthapadi, Samir Khuller, and An Zhu. Achieving anonymity via clustering. *ACM Trans. Algorithms*, 6(3), July 2010. doi:10.1145/1798596.1798602.

3 S. Ahmadian, A. Norouzi-Fard, O. Svensson, and J. Ward. Better guarantees for k-means and Euclidean k-median by primal-dual algorithms. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 61–72, October 2017. doi:10.1109/FOCS.2017.15.

4 Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Mach. Learn.*, 75(2):245–248, May 2009. doi:10.1007/s10994-009-5103-0.

5 Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999. doi:10.1145/304181.304187.

6 David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2007, pages 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics.

7 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004. doi:10.1137/S0097539702416402.

8 Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The Hardness of Approximation of Euclidean k-Means. In Lars Arge and János Pach, editors, *31st International Symposium on Computational Geometry (SoCG 2015)*, volume 34 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 754–767, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SOCG.2015.754.

9 Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Faster algorithms for the constrained k-means problem. *Theor. Comp. Sys.*, 62(1):93–115, January 2018. doi:10.1007/s00224-017-9820-7.

10 Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2011, page 26–40, USA, 2011. Society for Industrial and Applied Mathematics.

11 Jarosław Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median and positive correlation in budgeted optimization. *ACM Trans. Algorithms*, 13(2), March 2017. doi:10.1145/2981561.

12 Jaroslaw Byrka, Piotr Skowron, and Krzysztof Sornat. Proportional Approval Voting, Harmonic k-median, and Negative Association. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2018.26.

13 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From gap-eth to fpt-inapproximability: Clique, dominating set, and more. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 743–754, 2017.

14 Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002. doi:10.1006/jcss.2002.1882.

15 Ke Chen. On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009. doi:10.1137/070699007.

16 Vincent Cohen-Addad and Karthik C.S. Inapproximability of clustering in lp metrics. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS 2019)*, pages 519–539, 2019.

17 Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. Tight FPT Approximations for k-Median and k-Means. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2019.42.

18 Vincent Cohen-Addad, Philip N. Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in Euclidean and minor-free metrics. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS 2016)*, 00:353–364, 2016. doi:doi.ieeecomputersociety.org/10.1109/FOCS.2016.46.

19 Vincent Cohen-Addad and Jason Li. On the Fixed-Parameter Tractability of Capacitated Clustering. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 41:1–41:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2019.41.

20 Graham Cormode and Andrew McGregor. Approximation algorithms for clustering uncertain data. In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '08, page 191–200, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1376916.1376944.

21 Marek Cygan, MohammadTaghi Hajiaghayi, and Samir Khuller. LP rounding for k-centers with non-uniform hard capacities. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 273–282, 2012.

22 Sanjoy Dasgupta. The hardness of k-means clustering. Technical Report CS2008-0916, Department of Computer Science and Engineering, University of California San Diego, 2008.

23 Hu Ding. Faster balanced clusterings in high dimension. *Theoretical Computer Science*, 2020. doi:10.1016/j.tcs.2020.07.022.

24 Hu Ding and Jinhui Xu. A unified framework for clustering constrained data without locality property. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2015, page 1471–1490, USA, 2015. Society for Industrial and Applied Mathematics.

25 Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD 1996, page 226–231. AAAI Press, 1996.

26 Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the twenty-third annual symposium on Computational geometry*, SCG 2007, pages 11–18, New York, NY, USA, 2007. ACM. doi:10.1145/1247069.1247072.

27 Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a PTAS for k-means in doubling metrics. *SIAM Journal on Computing*, 48(2):452–480, 2019. doi:10.1137/17M1127181.

28 Dishant Goyal, Ragesh Jaiswal, and Amit Kumar. Streaming PTAS for constrained k-means, 2019. arXiv:1909.07511.

29 Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999. doi:10.1006/jagm.1998.0993.

30 Anupam Gupta and Kanat Tangwongsan. Simpler analyses of local search algorithms for facility location. *CoRR*, abs/0809.2554, 2008. arXiv:0809.2554.

31 Mohammadtaghi Hajiaghayi, Wei Hu, Jian Li, Shi Li, and Barna Saha. A constant factor approximation algorithm for fault-tolerant k-median. *ACM Trans. Algorithms*, 12(3), April 2016. doi:10.1145/2854153.

32 Erez Hartuv and Ron Shamir. A clustering algorithm based on graph connectivity. *Information Processing Letters*, 76(4):175–181, 2000. doi:10.1016/S0020-0190(00)00142-3.

33 Anil K. Jain, M Narasimha Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, September 1999. doi:10.1145/331499.331504.

34 Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC 2002, pages 731–740, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.510012.

35 Ragesh Jaiswal, Amit Kumar, and Sandeep Sen. A simple D^2-sampling based PTAS for k-means and other clustering problems. *Algorithmica*, 70(1):22–46, 2014. doi:10.1007/s00453-013-9833-9.

36 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, SCG 2002, page 10–18, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/513400.513402.

37 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 646–659, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188882.

38 Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2):5:1–5:32, February 2010. doi:10.1145/1667053.1667054.

39 Shi Li. Approximating capacitated k-median with $(1+\varepsilon)k$ open facilities. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, page 786–796, USA, 2016. Society for Industrial and Applied Mathematics.

40 Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, STOC 2013, page 901–910, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2488608.2488723.

41 Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is NP-hard. *Theoretical Computer Science*, 442:13–21, 2012. Special Issue on the Workshop on Algorithms and Computation (WALCOM 2009). doi:10.1016/j.tcs.2010.05.034.

42 Pasin Manurangsi. Tight running time lower bounds for strong inapproximability of maximum k-coverage, unique set cover and related problems (via t-wise agreement testing theorem). In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2020, page 62?81, USA, 2020. Society for Industrial and Applied Mathematics.

43 Pitu B. Mirchandani and Richard L. Francis. *Discrete Location Theory*. Wiley, 1990.

44 Latanya Sweeney. k-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002. doi:10.1142/S0218488502001648.

45 Andrea Vattani. The hardness of k-means clustering in the plane. Technical report, Department of Computer Science and Engineering, University of California San Diego, 2009.

46 J S Vitter. Random sampling with a reservoir. *ACM Trans. Math. Software*, 11(1):37–57, 1985.

47 Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2, August 2015. doi:10.1007/s40745-015-0040-1.